NASA Contractor Report 187592
ICASE Report No. 91-52

ICASE

MAPPING IMPLICIT SPECTRAL METHODS TO DISTRIBUTED MEMORY ARCHITECTURES

Andrea L. Overman
John Van Rosendale

Contract No. NAS1-18605
June 1991

Institute for Computer Applications in Science and Engineering
NASA Langley Research Center
Hampton, Virginia 23665-5225

Operated by the Universities Space Research Association

National Aeronautics and
Space Administration

Langley Research Center
Hampton, Virginia 23665-5225

Mapping Implicit Spectral Methods to Distributed Memory Architectures

Andrea L. Overman
NASA Langley Research Center
Hampton, VA 23665

and

John Van Rosendale*
Institute for Computer Applications in Science and Engineering
NASA Langley Research Center
Hampton, VA 23665

Abstract

Spectral methods have proven invaluable in numerical simulation of PDEs, but the frequent global communication required raises a fundamental barrier to their use on highly parallel architectures. To explore this issue, we implemented a three dimensional implicit spectral method on an Intel hypercube. Utilization of about 50% was achieved on a 32 node iPSC/860 hypercube, for a $64 \times 64 \times 64$ Fourier-spectral grid; finer grids yield higher utilizations.

Chebyshev-spectral grids are more problematic, since plane-relaxation based multigrid is required. However, by using a semicoarsening multigrid algorithm, and by relaxing all multigrid levels concurrently, relatively high utilizations were also achieved in this harder case. In fact, since the amount of work per processor was higher in this case, we achieved somewhat higher utilization, typically 60% on moderate sized problems. Thus spectral methods remain attractive on the current generation of distributed memory architectures.

*Research supported by the National Aeronautics and Space Administration under NASA Contract NAS1-18605, while the second author was in residence at ICASE.

1. Introduction. Spectral methods have proven of great value in numerical simulation of turbulence, numerical weather prediction, acoustics, and in a variety of other applications. Unlike difference methods, spectral methods accurately approximate all frequencies present on the grid, and are thus well suited to problems, like turbulence, where accurate resolution of evolving solutions is critical. However, on parallel machines, the frequent global communication required by spectral methods seems to impose a fundamental barrier to their continued use. In this paper, we study this issue, in the context of time dependent implicit spectral methods.

Communication arises in spectral methods in two principal ways. First, evaluation of the spectral operator requires global communication, usually in multidimensional FFTs. Second, in many problems, viscous stability limits constrain the time step so severely that one must resort to implicit methods. This holds true especially for Chebyshev-spectral methods, where the close spacing of collocation points near boundaries imposes severe stability limits on explicit methods.

Given the frequent global communication required, it is not clear whether spectral methods remain attractive on parallel architectures. That is, the subtle numerical advantages of spectral methods may be completely swamped by the high cost of communication and synchronization. This would be unfortunate, since there are many applications in which spectral methods are invaluable[2]. Thus we undertook to study the basic issues involved in implementing spectral methods on distributed memory machines, in order to assess the extent to which spectral methods remain competitive on these architectures.

To explore this issue, we designed and implemented an implicit spectral method on the Intel iPSC/860 hypercube. Our program performs a multigrid-based implicit solution of the time dependent, variable coefficient Helmholtz equation,

$$u_t \;=\; \nabla a(x,y,z)\cdot\nabla u \;-\; b(x,y,z)u \;+\; f(x,y,z,t),$$

on three dimensional tensor product grids, a problem arising in the Uzawa formulation of the incompressible Navier Stokes equations and in ocean circulation problems.

2. Finite Element Preconditioners. One can solve the nearly dense linear systems[2] arising in spectral methods in a variety of ways. One of the best approaches is to use a Richardson or conjugate gradient iteration, preconditioned by inversion of a low order finite element system. Suppose S is the Fourier-spectral discrete Laplacian, and let H be a low order finite difference or finite element operator. If H is the standard 5 point Laplacian in two dimensions, the spectral condition number of the preconditioned spectral system, $H^{-1}S$, is $\kappa = 2.47$, while for bilinear finite elements it is 1.44.

With this improved condition number, noted originally by Deville and Mund[4], the convergence rate of optimal parameter Richardson iteration, given by

$$\rho \;=\; \frac{\kappa \;-\; 1}{\kappa \;+\; 1}$$

drops from 0.42 to 0.18, making Richardson iteration quite effective.

The condition number of the preconditioned Fourier-spectral operator can be further improved by introducing mass lumping into the finite element discretization. In one dimension, this amounts to replacing the finite element system

$$K\ \mathbf{u} \;=\; M\ \mathbf{f}$$

by an analogous system with a partially lumped mass matrix:

$$\tilde{M} \;=\; 0.95\ M \;+\; 0.05\ I$$

In higher dimensions, one gets the same effect by tensoring one dimensional discretizations. Suppose one has the improved one dimensional finite element discretizations

$$\begin{aligned} K_x^j\ \mathbf{u}^j \;&=\; \tilde{M}_x^j\ \mathbf{f}^j \\ K_y^i\ \mathbf{u}^i \;&=\; \tilde{M}_y^i\ \mathbf{f}^i \end{aligned}$$

along x and y mesh lines respectively. Then the analogous two dimensional finite element discretization is:

$$(K_x^j \otimes \tilde{M}_y^i \;+\; \tilde{M}_x^j \otimes K_y^j)\ \mathbf{u} \;=\; \tilde{M}_x^j \otimes \tilde{M}_y^i\ \ \mathbf{f}$$

The condition number of the preconditioned spectral system, based on this improved finite element discretization, is 1.26, for a Richardson convergence rate of 0.11. Since one typically needs to reduce the initial residual by four or five orders of magnitude at each implicit time step, five or six preconditioned iterations are needed per time step.

3. Fourier-spectral Grids. Fourier-spectral codes are used in problems with periodic boundary conditions. For our model problem, at each time-step, we form spectral residuals by fast Fourier transforms, and invert the spectral linear system by a sequence of preconditioned Richardson iterations. Since the convergence rate of the Richardson iteration is 0.11, one multigrid V-cycle suffices to adequately solve the preconditioning finite element system at each iteration.

Data Distribution. The grids need to be distributed across the processors in a way that minimizes communication and balances the load. In this section, the communication requirements of two alternate data distributions, which we refer

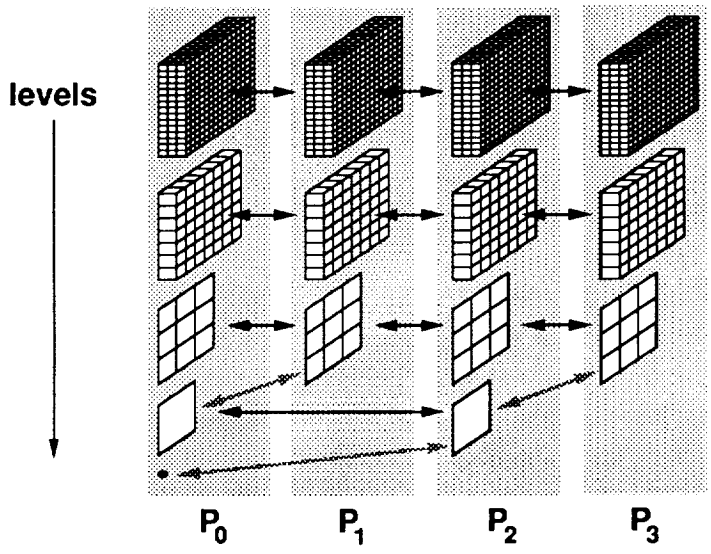

Figure 1: Multigrid Using the Z-Distribution

to as the z- and the xyz-distributions, are compared. In the z-distribution, only the z direction is blocked, so adjacent xy-planes are placed on each processor. In the xyz-distribution, all three coordinate directions are blocked, so each processor owns a subcube of the grid. Both data distributions force communication in both the FFT residual computation and in the multigrid solve.

With both distributions, we compute the FFTs in all three coordinate directions sequentially on processors, using global exchanges to bring all required data into the processors. Since each xy-plane lies in a single processor with the z-distribution, this distribution requires interprocessor communication only for the z-direction FFT. In this case, a complete exchange of data, where each processor sends a block of data of size n^3/p^2, for an $n \times n \times n$ grid, to the $p-1$ other processors, is required.

The xyz-distribution requires approximately twice as much communication for the spectral residual computation. Communication is required so that each processor holds xy-planes before the FFTs in the x and y directions are calculated. In addition, before the z direction is calculated, processors must hold yz or xz planes. Both exchanges can be done with each processor sending messages of size $n^3/p^{5/3}$ to $p^{2/3}-1$ other processors.

Both the z- and the xyz-distributions also require communication of boundary data after each relaxation, restriction, and prolongation in the multigrid V-cycle. In the z-distribution, processor i must communicate boundary planes

to processors $i+1$ and $i-1$. In the xyz-distribution, each processor must communicate boundary data to six neighboring processors, but the message sizes are shorter. There is also additional interprocessor communication required with both data distributions, when performing restriction or prolongation operations between levels having idle processors. Since there are fewer active processors on each coarser level, the work of processors becoming idle needs to be sent to the remaining active processors.

For an $n \times n \times n$ problem, the number of grid levels in the multigrid V-cycle is $\log_2 n$. The number of grid points per level is 8^l, where l denotes the level number, with $l = 1$ the coarsest grid. The number of grid points owned by each processor, on the grid levels with no idle processors, is $8^l/p$ for both data distributions. The number of coarse grid levels containing idle processors for the z-distribution is determined by $\log_2 p$. On these levels, active processors contain one xy-plane with size 4^l. In the xyz-distribution, the number of coarse grid levels containing idle processors is determined by $\frac{1}{3}\log_2 p$, and on these levels, active processors contain only one grid point. Figure 1 shows the communication required by the z-distribution during the multigrid V-cycle.

Using this information, we computed the amount of communication required per Richardson iteration for each of the data distributions. The message startup and byte transfer rate were estimated using values reported for the iPSC/860 by Bokhari[1]. Data distribution in the z-direction led to higher communication costs in the multigrid algorithm, due both to the greater load imbalance and longer message lengths. However, the additional communication required by the xyz-distribution for the FFTs led to higher communication costs for the complete Richardson iteration. The amount of communication is a function of both problem size and the number of processors. Generally, for large n and $p \approx n$, the xyz-distribution is more efficient, while for moderate n and p, the z-distribution is superior. Based on this analysis, we implemented the Fourier-spectral code using the z-distribution, since it appeared to be significantly more efficient for our machine and problem sizes.

Experimental Results: Fourier Case. The experiments were performed on the 32-processor iPSC/860 at ICASE. Using the best current compiler (Portland Group), our utilization was about 65% for the spectral residual calculation and about 40% for the multigrid solution. The load imbalance and large number of communication steps in multigrid led to this lower utilization. Thus our overall utilization, including both the residual calculation and multigrid solution was about 50%.

4. Chebyshev Grids. Solving the spectral equations on Chebyshev grids is inherently more difficult, since the grid stretching leads to poor condition numbers, and since the matrix corresponding to the pseudospectral discretization

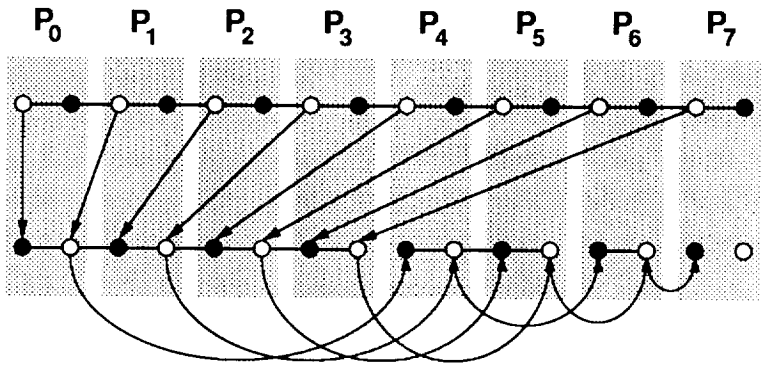

Figure 2: Concurrent Multigrid Data Distribution

on Chebyshev grids is asymmetric. However, a more serious issue is the problem of solving the finite element system on highly stretched grids. When the mesh aspect ratios

$$\Delta x/\Delta y, \qquad \Delta y/\Delta x$$

are large, point relaxation multigrid is ineffective. Line relaxation suffices to resolve this problem in some cases; however, in the general case, one must use plane-relaxation based multigrid.

There are two viable kinds of plane-relaxation based multigrid, algorithms employing plane relaxation sweeps in all three coordinate directions and algorithms using plane relaxation in only one direction. The latter are known as "semi-coarsening" algorithms, since the grid is coarsened in only one coordinate direction. That is, if the fine grid is an $n \times n \times n$ grid, the next coarser grid will be an $n \times n \times n/2$ grid, the one after that will be an $n \times n \times n/4$ grid, and so on.

Semi-coarsening algorithms are cheaper than plane relaxation algorithms with relaxation in all three coordinate directions, since plane relaxation is needed in only one direction. They also converge faster and are less sensitive to grid stretching[3]. In addition to these numerical advantages, this algorithm is attractive for parallel computing, since the z-distribution allows the plane relaxations to be carried out with relatively little interprocessor communication.

Despite these advantages, there is an inherent problem with this approach;

the sizes of the planes do not decrease as one goes to coarser grids, leading to very poor utilizations. We addressed this issue by using concurrent iteration in which all grid levels are simultaneously relaxed[5]. Combining a concurrent relaxation multigrid algorithm in the z-direction, with a standard semi-coarsening line relaxation algorithm in xy-planes, led to a robust and effective algorithm which is highly parallel and maps easily to distributed memory architectures.

Experimental Results: Chebyshev Case. The Chebyshev multigrid was implemented using a concurrent iteration multigrid scheme with red-black plane relaxation. Figure 2 illustrates the distribution of fine and coarse grid planes across the processors and the communication required during the restriction phase. The prolongation communication is similar to that for the restriction. For the Chebyshev case, we obtained processor utilization of approximately 60% for a $32 \times 32 \times 32$ problem. The increase in utilization was due to both the improved load balance and the increased ratio of computation to communication.

5. Conclusions. Mapping implicit spectral codes to distributed memory architectures is difficult. While we achieved 50% processor utilization on both the Fourier-spectral and Chebyshev-spectral codes, this performance is very sensitive to the architecture's communications capabilities. If processor speeds were to increase by a factor of ten, without a commensurate increase in communication bandwidth, spectral methods would become virtually unusable.

As can be seen from our results, we obtained reasonable processor utilization, despite the relatively small size of problems considered, without extensive program "tuning." With larger problems, having perhaps 1024 mesh points in every direction, we would expect to achieve 75% processor utilization on hypercubes having a few thousand processors, assuming the present communication/computation speed ratio. The amount of exploitable parallelism on this class of applications is really very large.

To achieve high utilization on machines having thousands of processors will require several improvements in our algorithm. First, some level of overlap of communication and computation is necessary. While this is trivial in principle, it entails extensive programming changes. Second, alternate ways of distributing the computation on the hypercube needed to be explored. While our approach of distributing the data in only the z-direction is optimal in some cases, it exacerbates the multigrid "idle processor" problem on coarse grids and increases total communication. Thus it may be better to use hybrid decompositions, in which some grid levels are decomposed in one way and others in other ways. Third, new variants of multigrid[7], based on the use of multiple coarse grids[1]

[1]W. Hackbusch is also exploring the use of multiple coarse grids to obtain robustness (personal communication).

promise to eliminate the need for line and plane relaxation altogether, allowing much higher levels of parallelism. This is probably the most fruitful direction for future research in this area.

Exploring these issues is interesting, but rather awkward at the moment, with the current Intel software environment. The availability of better programming environments, such as the Kali, Dino, and Fortran D languages[6, 8] should dramatically ease exploration of such alternatives.

References

[1] S. Bokhari. *Communication overhead on the Intel iPSC-860 hypercube.* Technical Report ICASE Interim Report 10, ICASE, 1990.

[2] C. Canuto, M.Y. Hussaini, A. Quarteroni, and T.A. Zang. *Spectral Methods in Fluid Dynamics.* Springer-Verlag, 1988.

[3] N. H. Decker and J. Van Rosendale. *Operator induced multigrid algorithms using semirefinement.* In J. Mandel, et. al., editor, Proceedings of the Fourth Copper Mountain Conference on Multigrid Methods, pages 87–105. SIAM, 1989.

[4] M.O. Deville and E.H. Mund. *Chebyshev pseudospectral solution of second-order elliptic equations using finite element preconditioning.* J. Comput. Phys., 60(517), 1985.

[5] D. Gannon and J. Van Rosendale. *On the structure of parallelism in highly concurrent pde solvers.* Parallel and Distributed Computing, 3:106–135, 1986.

[6] P. Mehrotra and J. Van Rosendale. *Parallel language constructs for tensor product computations on loosely coupled architectures.* In Proceedings Supercomputing '89, pages 616–626, November 1989.

[7] N. Naik and J. Van Rosendale. *Robust parallel multigrid using multiple semi-coarse grids.* Preliminary Proceedings of the Fifth Copper Mountain Conference on Multigrid Methods, 1991.

[8] M. Rosing and R. Schnabel. *An overview of DINO - a new language for numerical computation on distributed memory multiprocessors.* Technical Report CU-CS-385-88, University of Colorado, Boulder, 1988.

Report Documentation Page

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.
NASA CR-187592 ICASE Report No. 91-52		

4. Title and Subtitle	5. Report Date
MAPPING IMPLICIT SPECTRAL METHODS TO DISTRIBUTED MEMORY ARCHITECTURES	June 1991
	6. Performing Organization Code

7. Author(s)	8. Performing Organization Report No.
Andrea L. Overman John Van Rosendale	91-52
	10. Work Unit No. 505-90-52-01

9. Performing Organization Name and Address	11. Contract or Grant No.
Institute for Computer Applications in Science and Engineering Mail Stop 132C, NASA Langley Research Center Hampton, VA 23665-5225	NAS1-18605
	13. Type of Report and Period Covered: Contractor Report

12. Sponsoring Agency Name and Address	14. Sponsoring Agency Code
National Aeronautics and Space Administration Langley Research Center Hampton, VA 23665-5225	

15. Supplementary Notes

Langley Technical Monitor:
Michael F. Card

To appear in the Proc. 5th SIAM Conf. on Parallel Computing, Houston, TX, March 1991.

Final Report

16. Abstract

Spectral methods have proven invaluable in numerical simulation of PDEs, but the frequent global communication required raises a fundamental barrier to their use on highly parallel architectures. To explore this issue, we implemented a three dimensional implicit spectral method on an Intel hypercube. Utilization of about 50% was achieved on a 32 node iPSC/860 hypercube, for a 64 x 64 x 64 Fourier-spectral grid; finer grids yield higher utilizations.

Chebyshev-spectral grids are more problematic, since plane-relaxation based multigrid is required. However, by using a semicoarsening multigrid algorithm, and by relaxing all multigrid levels concurrently, relatively high utilizations were also achieved in this harder case. In fact, since the amount of work per processor was higher in this case, we achieved somewhat higher utilization, typically 60% on moderate sized problems. Thus spectral methods remain attractive on the current generation of distributed memory architectures.

17. Key Words (Suggested by Author(s))	18. Distribution Statement
Spectral methods, parallel multigrid	64 - Numerical Analysis Unclassified - Unlimited

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of pages	22. Price
Unclassified	Unclassified	9	A02

NASA FORM 1626 OCT 86